Towards Accreditation of Diagnostic Models for Improved Performance

Anuradha Kodali[1], and Peter Robinson[2]

[1]*SGT Inc., NASA Ames Research Center, Moffett Field, CA, 94035*
anuradha.kodali@nasa.gov

[2]*NASA Ames Research Center, Moffett Field, CA, 94035*
peter.i.robinson@nasa.gov

ABSTRACT

The research community mainly concentrates on developing new and updated diagnostic algorithms to achieve high diagnostic performance which is necessary but not sufficient for the diagnostic models that are embedded in software. The focus of this paper is to understand the requirements for accrediting diagnostic system models to meet high performance and safety criticality in case of both models and embedded system (model + software). For embedded systems, models need to be accredited first to allow a more accurate distinction of whether the model or the code within which the model is embedded is the cause of degraded performance. This is because, neither standards for models and simulations (NASA-STD-7009) nor software engineering requirements (NPR 7150.2A) are sufficient to accredit the models in embedded systems. NASA-STD-7009 assesses the correctness of the physics in models and simulations and NPR 7150.2A lists software engineering requirements for NASA systems. Thus, it is important to understand the accreditation standards in terms of performance requirements of models in embedded systems that can smoothly transit from NASA-STD-7009 to NPR 7150.2A. We will discuss interactive diagnostic modeling evaluator (i-DME) as an accreditation tool that provides the performance requirements or limitations imposed while accrediting embedded systems. This process is done automatically, making accreditation feasible for larger diagnostic systems.

1. INTRODUCTION

The research community over prior years has concentrated on developing new and updated diagnostic algorithms to avoid diagnostics with ineffective reasoning. But, most of the times, the real root cause of this ineffectiveness is attributed to incomplete or inaccurate diagnostic models (Simpson, & Sheppard, 1991). The models are incomplete due to the constraints arising from cost (e.g. test design) and system complexity issues. Importantly, with increasing complexity, detailing and bookkeeping of the system becomes very difficult leading to missed information in diagnostic models (Sheppard, & Simpson, 1993). Secondly, the models can be inaccurate because of the following reasons: 1. lack of technical expertise, 2. misunderstanding the existing expertise (documents), and 3. human errors. While human errors are unpredictable; the others can be resolved by precise planning and better documentation at every step of model development. Especially, the first two reasons are categorized as novice and intermediary levels of human knowledge, respectively; but even experts can make errors.

Traditionally, diagnostic modeling is independent of design and manufacturing (Simpson, & Sheppard, 1991). Diagnostic modelers build their models by studying design documents and technical manuals. Here, the physics model is fixed while building diagnostic models and optimizing it for maximum performance. Hence, in early 1980s, there was a strong drive to include diagnostics as an engineering task during system development. For this purpose, testability analysis is strategized to include adding/modifying tests, repacking components to decrease ambiguity, decreasing false-alarms, and improving the observability of certain faults (Simpson, & Sheppard, 1992). Testability analysis, while included in system development, decreases maintenance cost and time, and also improves efficiency of diagnostic models without disturbing system's operational performance by supporting sensor selection and placement.

But, the testability methodology ignores three salient features. Firstly, determining fault modeling (at what level), and the causal relationship between faults and tests are not included for testability analysis. Secondly, while performing testability, the diagnostic algorithm is not included to assess

the diagnostic performance; thus there is no remedy for misdiagnosis that is incurred later. Thirdly, no cost-effective repair procedure for the system/diagnostic model is provided. Thus, the best strategy here is to verify and validate the diagnostic model by analyzing all its characteristics (faults, tests, etc.) and inserting the faults via simulation to assess the diagnosis (Sheppard, & Simpson, 1998). Considering these factors, Interactive Diagnostic Modeling Evaluator (i-DME) (Kodali, Robinson, & Patterson-Hine, 2013) is developed as an automatic computer-user interactive tool that proposes cost-effective repair strategies related to fault modeling, test design, and their relationship. This is performed on the D-matrix (Luo, Tu, Pattipati, Qiao, & Chigusa, 2006), an abstract representation of the diagnostic model with causal fault-test relationship in terms of 0's and 1's. Matrix entry 1 represents that the test detects the corresponding fault, otherwise vice-versa. Note that adding/removing tests needs changes in both system and diagnostic models. For the other repairs pertained only to diagnostic models, they can be performed even after system development. But, this is not advisable because the diagnostic models will be implemented in software before the end of system development and it is not easy modifying the software always. Note that software is required to implement the diagnostic models and it is important to certify both the model and software for the same required output.

Columbia Accident Investigation Board (CAIB, 2003) stresses the accreditation (certification) of embedded systems (model + its implementation software, for e.g. TEAMS Designer, TEAMS-RDS (Qualtech Systems Inc.)) to "develop, validate, and maintain physics-based computer models (models in embedded systems)". This process is different from accrediting the models alone. These models are pre-accredited before certifying the embedded system. Such a distinction is important to find out if the model or the code is the cause for degraded performance. For this purpose, we are working to achieve the NASA accreditation standards for models and simulations (NASA-STD-7009), and software engineering requirements (NPR 7150.2A) to make them suitable for embedded systems. But, unfortunately, neither of these standards independently, or combined can provide the necessary standards for all the model-based embedded systems. Clearly, the requirements from models that should be satisfied by the embedded system, the inputs to the accreditation requirements of the software code which implements the model, and the relationship of the model and the code accreditation results needs strict scrutiny and is the focus of this paper. This process is also helpful to not expect from the code performance beyond the limitations of its embedded model. This process becomes tedious with large-scale diagnostics models. Thus, it is important to automatically generate the accreditation requirements to the embedded system via interactive diagnostic modeling evaluator (i-DME). This tool repairs the diagnostic models for better diagnostic performance and then certifies them. As a result the necessary requirements are derived for the diagnostic model's implementation in embedded systems.

Thus, this paper details the general performance guidelines for diagnostic models and the corresponding accreditation process when implemented in software. In Section 2, we will address the building of diagnostic models and best modeling practices. We will also explain i-DME architecture's potential as a model accreditation tool. This tool automatically provides necessary standards information to accredit models implemented in embedded system, thus makes it easier to accredit larger diagnostic models. The NASA standard for models and simulations, and software engineering requirements and their interconnection are studied in order to perform accreditation for embedded systems in Section 3. We will summarize the findings in Section 4.

2. MODELING OF SYSTEM AND DIAGNOSTIC MODELS

In a natural sequence of development, diagnostic modeling follows the system development in parallel. Later, the diagnostic model is implemented in software (embedded systems). It is important to have best practices at every phase of development for the required performance. In this section, we focus on system and diagnostic model development and the corresponding tool (i-DME) to enable best accreditation practices for better diagnostic performance.

2.1. System Modeling

System modeling is an important engineering task which requires adequate planning and skillful implementation. Here, modeling includes developing a combination of conceptual, mathematical, logical and/or computational models. Firstly, the personnel in charge of modeling starts with the specifications required to satisfy the objectives and the mission. Then, the conceptual designs are translated into detailed developmental plans for the molding of hardware. At this stage, the personnel in charge can change the requirements set before to suit practical compulsions. This may lead to changing the basic principles and to refine the existing methods continuously. After this, there will be extensive testing, both manually and through test development, to shift the development into qualification – once simulated and real time series data is available. There will be two types of tests: development tests to verify the components to consistently and reliably perform; quantification tests to determine if the vehicle is suitable to perform its specified mission. The system is intensively verified and validated by detecting design deficiencies and early development failures arising from the unanticipated communication among components. This process includes verifying for authenticity of operating conditions, e.g.

pressure and temperature and efficiency of each component/subsystem performance. The validation testing strategy focusses to build a system that is effective and economically viable. This means the model can be built in a timely fashion within the budget structure that accomplishes the mission objectives (Swenson, & Grimwood, 1989).

2.2. Diagnostic Modeling

While system development ensures design for performance; it is important to design it for field operations via an optimized diagnostic model (Simpson, & Sheppard, 1993). The diagnostic information is extracted from system models via technical manuals and design documents. This knowledge is then used to specify a simplified form of the diagnostic model; this is used for testability analysis and diagnosis later. Even though the system development phase is well documented; building diagnostic models as a separate task is troublesome. By doing this routine as part of system development; time, cost, and efficiency of diagnosis can be improved simultaneously without hindering the system's operational mechanism. For e.g. designing tests early to decrease ambiguity at individual, sub-system, and system levels reduces maintenance cost (Simpson, & Sheppard, 1992) (Sheppard, & Simpson, 1992). Also, via this process, the personnel are forced to not only think about performance, but also focus to recover it from a failure condition. This paper once again advocates practicing diagnostic modeling within system development; thus analyzing the system for diagnosability and testability from its early stages of development.

2.2.1. Fault Modeling and Test Design

The first important task in fault modeling is to determine the level at which the diagnosis is performed (Simpson, & Sheppard, 1992). It can be done at component, or sub-system, or system level. In general the level to which diagnostics should be performed is the level to which repair actions can be taken (e.g. LRU – line replaceable unit, ORU- orbital replacement unit). The symptoms associated with each fault mode are analyzed during FMECA analysis (Sheppard, & Simpson, 1992). The corresponding impact, in terms of criticality of the fault mode on mission success, safety, system performance, maintainability, and maintenance requirements is also analyzed. Correspondingly, tests are designed to detect these faults. High detection and design costs are always considered during test design. Also, with the fault dictionary (D-matrix); the set of dependencies between the tests and the fault modes are determined via simulations, dataflow analysis, logic flow analysis, and traditional, manual circuit analysis (Sheppard, & Simpson, 1992) (Luo, Tu, Pattipati, Qiao, & Chigusa, 2006).

When diagnostic models are optimized for better performance from early stages of system development; analysis for fault mode definitions and optimized test designing is performed. This includes analyzing the model for ambiguity in fault modes and designing tests to reduce it. Similarly, analysis for excess (excess test provides the same information as a combination of other tests (Simpson, & Sheppard, 1992)) and redundant tests is performed by incorporating only essential tests that are required for diagnostics. Instead of restricting the tests to check for proper system functioning; they are also required to isolate faults in the model. Models are made up of nodes and arcs, and the propagation paths for fault models are complicated for a complex system. So, it is always important to carefully generate the fault-test relationship in D-matrix. With addition of new components during system development, this relationship is bound to change and should be updated accordingly.

2.2.2. Accreditation of Diagnostic Models: i-DME

Diagnostic modeling has matured from a simple data and file sharing to computerized automatic designer tools (e.g. TEAMS (Qualtech Systems Inc.)). This necessitates accreditation of diagnostic models and their real-time software implementation. The aim is to reduce mean time to isolate faults and recover systems with highest efficiency (Simpson, & Sheppard, 1991). But, this may not always be the case because of improper understanding of testability information. Certain measures (e.g. ambiguity, operational fault isolation etc.) are extracted from the model to check for testability and accordingly, the systems are redesigned (at initial stages) or repackaged (Sheppard, & Simpson, 1992). Similarly, we have focused on building new tests for improved performance of the diagnostic model in isolating faults. But, adding tests is not always the sufficient solution because it may cause other issues with the system operation and cost effectiveness. This debugging and remedial process is always tedious and is impossible for human efforts. Thus, in the realm of system engineering, i-DME tool is developed to debug diagnostic models at every step of system development and operation. This tool, with the aid of supervised data (data is labeled with corresponding nominal or faulty state), debugs diagnostic models and proposes repair strategies to D-matrix (abstract representation of diagnostic model) by coordinating with the decision maker (user) (Kodali, Robinson, & Patterson-Hine, 2013).

i-DME is defined as a combined process of computer and user decisive mechanisms where computer provides platform of the diagnostic analysis of the system model with the aid of supervised data and the decision maker performs the role of accepting/declining repair strategies based on the analysis of performance metrics and technical expertise (see Figure 1). Five D-matrix repair strategies are identified arranged in ascending order of cost effectiveness. These strategies range from addressing duplicity in faults and tests, repairing the fault universe to accommodate lower/higher level fault modeling (re-define the level of fault modeling

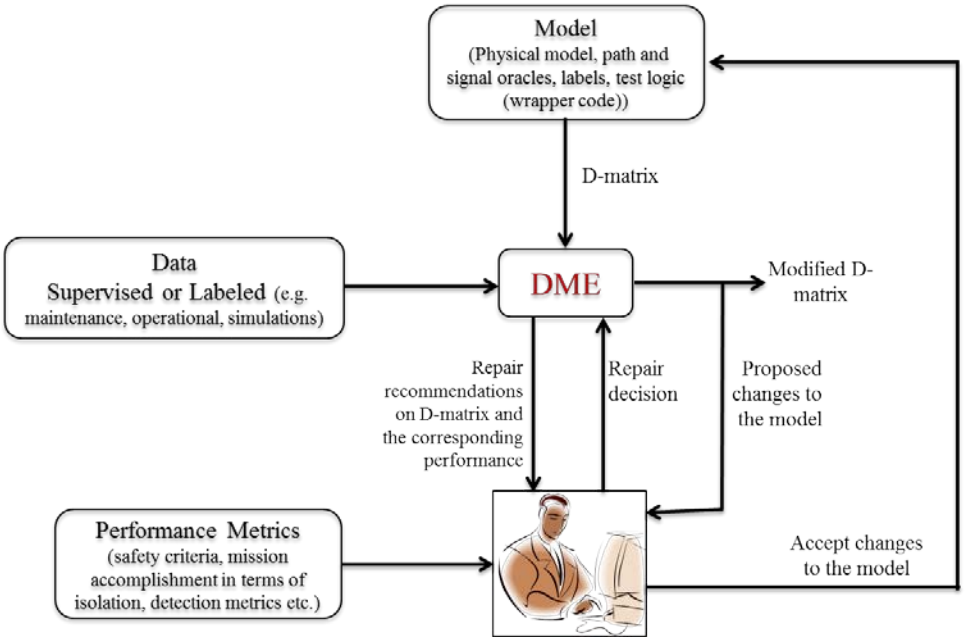

Figure 1. i-DME architecture

by adding or removing rows), repairing/changing the wrapper/test logic, repairing 0's and 1's in the D-matrix entries, and adding/removing tests. They are included in an iterative loop to experiment for better performance along with the decision maker. The performance criteria are based on fault detection and isolation metrics derived from the mission objectives by the user. Then, the decision maker accepts/declines the repair strategies based on before and after performance. More details of this framework can be found in (Kodali, Robinson, & Patterson-Hine, 2013).

In this process, the user not only plays a key role to accept/decline the repair on the diagnostic model, but also prepares the supervised data. The data collected via simulations, maintenance, or operations should be labelled with either nominal or the faulty condition. The credibility of the data depends on skill level of the user. The data can be used to validate the diagnostic model in i-DME process[1]. The system realities which cannot be formalized are also included as user's technical knowledge. Similarly, any diagnostic algorithm which will be employed for diagnosis during operations is implemented in this process for assessing the performance by calculating the corresponding metrics. Importantly, the diagnostic algorithm implemented here for diagnosis is also employed in the software implementation of the system during operations[2].

[1] i-DME efficiency is directly related to the authenticity of the supervised data used for accreditation.

[2] Presently, i-DME is explained for D-matrix; but the framework will be well extended to other modeling paradigms (e.g. fault signature matrix generated using temporal causal graphs (Daigle, Roychoudhury, Biswas, & Koutsoukos, 2010)).

i-DME as an accreditation tool

i-DME not only debugs diagnostic models, but can also double as an accreditation tool for diagnosis and proposes repair strategies to suit the performance. The salient features of this model accreditation tool are listed here:

1. The tool tracks the repairs and diagnostic performance of the diagnostic model throughout the system development and operations, and thus provides important inputs of the performance trends with each repair for higher diagnosability to the modelers (verification and validation).
2. The tool in addition to pointing out the errors or incompleteness in the model provides the strategies about what to do in order to improve the performance.
3. The requirements for system's accreditation are always specified in terms of operation and safety. But, in addition, this tool introduces and derives system requirements in terms of diagnostic performance, viz. detection and isolation metrics when analyzing diagnostic models by including diagnostic reasoning algorithm. It is especially useful to understand the limitations of cost of diagnostic modeling vs performance.
4. The tool adds value by utilizing the advantages of both computer and the decision maker, propose cost-effective repairs that not only include adding/modifying tests, but also corrects the level of fault modeling and causal fault-test relationship; thus investigating all the possible causes of erroneous models.

3. NASA STANDARDS: BRIDGING GAP BETWEEN MODEL AND SOFTWARE ACCREDITATION

In the prior discussion, accreditation process is performed on the models alone by proposing repairs for better diagnostic performance. But, it is also important to certify the embedded systems they are implemented in. This is because, in such a case, it is hard to distinguish if the performance degradation is due to error or incompleteness of the model or software in which it is embedded. For this purpose, test evaluation and execution are evaluated automatically in contrast to the regular practice by hand for software testing (Vaandrager, 2006). The response for each test case is noted when analyzing model against which the embedded system can be tested (Sabetzadeh, Nejati, Briand, & Mills, 2011).

In NASA's context, it is natural to think that the integration of NASA-STD-7009 for models and NPR 7150.2A for software engineering would provide the guidance that is required to accredit embedded diagnostic models. But, to date there is much ambiguity in guidance to accredit embedded model-based systems. In this paper, we focus on accrediting a subset of those systems, viz. diagnostic models.

NASA-STD-7009 provides methods to accredit models, but explicitly states that it does not apply to models and simulations that are embedded in control software, emulation software, and stimulation environments. It also points to NPR 7150.2A, NASA software engineering requirements to apply for such embedded models and simulations. But, in NPR 7150.2A, numerical accuracy, uncertainty analysis, sensitivity analysis, verification and validation for software implementation of models and simulations are stated to be addressed by the center processes and explains that the specific verification and validation information is available in NASA-STD-7009. This is in fact very confusing because NASA-STD-7009 doesn't apply to models and simulations implemented in certain embedded systems. Even for others, as specified in requirements mapping matrix of NPR 7150.2A, models are accredited as per this standard only when they support qualification of flight operations or equipment and ignores for e.g. ground operations/equipment for medium-critical systems (requirement SWE-070 in NPR 7150.2A).

The NASA Software Engineering Handbook (Section 7.15) (NASA software engineering handbook, 2013) recognizes this lack of specific direction and provides additional guidance which states that the analysis of models not covered by NASA-STD-7009 should report requirements 4.2.6, 4.4.1-4.4.9 found in NASA-STD-7009 while implementing NPR 7150.2A. It goes on to state that it is sufficient to merely report on any and all activities performed even reporting that no activities were performed.

For other models, it is important to ensure that the requirements of both the standards (NASA-STD-7009 and NPR 7150.2A) are satisfied. The requirements of NPR 7150.2A are either supplemental, or not related, or subset to the requirements in NASA-STD-7009. In either case, it is important to identify and derive the requirements from the diagnostic models that can be imposed on its embedded implementation. Hence, the process of accrediting embedded diagnostic systems includes 2 tasks: 1) identify the requirements for the accreditation of embedded systems, 2) implement an automated process (i-DME) to derive the requirements (in terms of performance requirements and reports) from the diagnostic model analysis.

3.1. Task 1: Identify the Accreditation Requirements

It is important to identify the input requirements from the model accreditation (NASA-STD-7009) that should be satisfied by the embedded system. This includes documenting the limitations of the model, conceptual details and rationale of the model and test cases, error and warning reports, and credibility scale for the eight assessment factors. The requirement extracted from these documents set the additional new performance requirements for the embedded system. Then, the relationship between the model and the code accreditation results should be scrutinized. This comparison for a similar set of test cases will check if the model is correctly implemented in the software or not. To this effect, we will explain the necessary information derived from model accreditation to the embedded system.

The verification and validation requirements of models as stated in NASA-STD-7009 required for embedded systems are listed as below:

1. Req. 4.4.1 – Shall document any verification techniques used and any domain of verification (e.g., the conditions under which verification was conducted).
2. Req. 4.4.2 – Shall document any numerical error estimates (e.g., numerical approximations, insufficient discretization, insufficient iterative convergence, finite-precision arithmetic) for the results of the computational model.
3. Req. 4.4.3 – Shall document the verification status of (conceptual, mathematical, and computational) models.
4. Req. 4.4.4 – Shall document any techniques used to validate the M&S for its intended use, including the experimental design and analysis, and the domain of validation.
5. Req. 4.4.5 – Shall document any validation metrics, referents, and data sets used for model validation.
6. Req. 4.4.6 – Shall document any studies conducted and results of model validation.

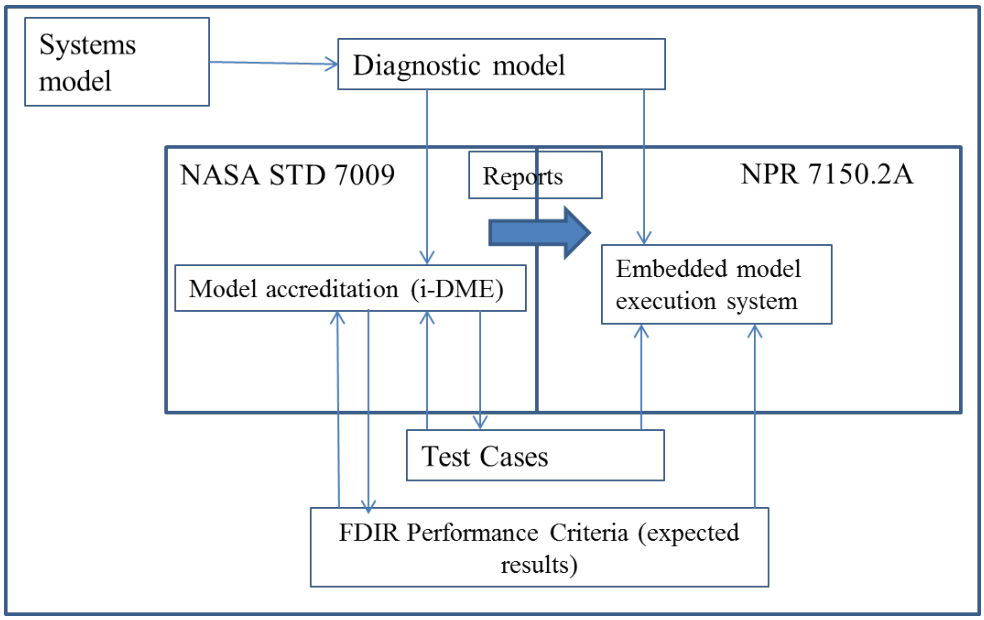

Figure 2. Relationship of i-DME to support 7009/7150.2A integration for embedded diagnostic models

The verification and validation information derived based on these requirements guides the accreditation process of embedded system of how to use and verify the model. Test cases that can be used for accreditation of both model and embedded system are defined and documented (see Figure 2). Similarly, verification and validation techniques (in this case, diagnostic algorithm) need to be the same for both accreditations and should be documented. Using this standard, the diagnostic model is independently accredited and the results are properly documented. In fact, every detail is documented as it is necessary to document everything that is performed or even document that nothing is done.

Analyzing the credibility of the model accreditation process is important to accredit embedded systems. To monitor this, NASA-STD-7009 has a credibility assessment score which is the weighted addition of eight factors, viz. verification, and validation (development), input pedigree, results uncertainty, and results robustness (operations), use history, management, and people qualifications (supporting evidence). These factors scored between 0 and 4 with 4 being the highest score. For e.g. input pedigree gets the highest score when the supervised data mimics the real-world operational data and captures all the necessary problems of interest. Similarly, the decision maker with extensive experience in the use of the diagnostic model corresponds to highest score for people. It is technically feasible, but with difficulty to achieve highest rating and is limited only when the system is in operation, while lower levels can be achieved during early phases of development. The credibility assessment score is documented and reported to the decision maker so that he understands the reliability of the model accreditation results.

Reporting errors and warnings is also a necessary requirement to translate the information from model to embedded system accreditation. During accreditation of diagnostic models; if it is identified that certain repairs to the model cannot be performed due to cost or complexity constraints, then, we document it as a constraint on the performance requirements of the embedded system. Otherwise this deficiency can be attributed to the code while it is being accredited. For e.g. information about components that are not diagnosable with the present model should be documented so that when it is not diagnosed with the working software; wrong manifestation to software can be avoided.

3.2. Task 2: i-DME to Generate Accreditation Requirements

For models of large-scale complex systems, the reporting of the requirements is a huge burden. In addition no specific model assurance activity processes are defined which makes it impossible with laborious manual labor to document the verification and validation requirements. This gap is filled in by the proposed method, i-DME that automatically generates reports for verification and validation requirements in NASA-STD-7009 as stated above. In addition, most importantly, i-DME defines the performance requirements that need to be and can be satisfied by the embedded system derived from the diagnostic model analysis.

The reports for these requirements will be accomplished by running i-DME system on a set of test cases which cover the potential failure sources in the system. For this purpose, as shown in Figure 2, the inputs for model verification and

validation are supervised test cases and user-set performance requirements. Using these, i-DME verifies and validates the diagnostic model by proposing repairs to add new failure modes/tests, or repair the test logic, or repairs the relationship between failure modes and tests in terms of 0's and 1's. After finishing the repair procedure, i-DME assesses the performance and changes the user-set performance criteria to a more realistic assessment. This acts as performance requirement to embedded systems. Similarly, i-DME in coordination with the user develops new test cases or makes corrections to the existing ones when the corresponding labels of nominal or off-nominal conditions are mistaken. All these requirements, test cases, and performance, are in line with those in NASA-STD-7009 and are documented in a user-friendly manner by the i-DME. The details about the diagnostic algorithm used for performance assessment will also be provided because it is mandatory to use the same technique while accrediting the model and the embedded system.

The capabilities of i-DME in the context of NASA-STD-7009 and NPR 7150.2A for the accreditation of models are listed below:

1. i-DME is an automated performance reporting tool. Thus, it becomes easier to accredit even very large scale diagnostic systems.
2. i-DME provides a framework to benchmark the diagnostic models against supervised data ("test cases"). These same test cases will also run against the code.
3. For verification and validation, the diagnostic algorithm calculates the performance in terms of detection and isolation metrics. This is also used to assess the credibility of the models for accreditation.
4. The system's faulty behavior as assessed by the diagnostic model is reported to the decision maker on a regular basis.
5. The limitations of the diagnostic model, for e.g. cannot achieve 100% isolation with insufficient tests, are obtained via i-DME process through the reporting to the decision maker. This avoids imposing incorrect performance requirements while accrediting embedded systems.

Conclusively, the diagnostic models and simulations are pre-accredited based on NASA-STD-7009 and then accredit the embedded system based on NPR 7150.2A by automatically deriving necessary requirements via i-DME. This enables clear distinction of the reason for performance degradation even in large-scale embedded systems. Also, by doing this, we understand what not to expect from the embedded system beyond the capabilities of the implemented model. This is because these limitations can be manifested as erroneous implementation in the code. Note that, diagnosing for errors in software code is not the focus of this paper.

3.3. Accreditation Requirements for ADAPT System

We demonstrate i-DME framework as an accreditation tool on ADAPT system (Poll, Patterson-Hine, Camisa, Garcia, Hall, Lee, Mengshoel, Neukom, Nishikawa, Ossenfort, Sweet, Yentus, Roychoudhury, Daigle, Biswas & Koutsoukos, 2007). During accreditation of D-matrix using i-DME framework, repairs are proposed to the D-matrix entries corresponding to voltage and current sensors of component FAN (underspeed and overspeed failure modes) to avoid misdiagnosis. This process is already published in (Kodali, Robinson, & Patterson-Hine, 2013) and is not presented here.

The information derived from ADAPT model accreditation needs to be reported for embedded system accreditation. The user sets correct isolation rate as the performance requirement on the model. Correct isolation rate is the percentage number of events that are correctly diagnosed (both nominal and faulty cases) over time. This metric is reported for each failure mode and nominal case whenever supervised data is available (see Figure 3). Note that, the performance requirement is based on user's decision and i-DME analyzes the model based on that metric. The diagnostic algorithm used during model accreditation, DMFD algorithm (Singh et al., 2009) is also reported. i-DME reports the performance requirements for embedded system accreditation as shown in Figure 3. The performance details (correct isolation rate) for each failure mode and nominal conditions against the given test cases along with the repair conditions proposed to achieve the corresponding performance are reported. These metrics are used to set requirements for comparison check for the available test cases when the software implementation of ADAPT diagnostic model is accredited.

4. CONCLUSIONS

In this paper, the accreditation process for diagnostic models and the corresponding embedded systems is discussed. It is important to include building of diagnostic models during system development so that any changes to the system model for better diagnosability can be proposed early. In this perspective, to debug diagnostic models at every step of development and operations, i-DME tool can be employed. As an accreditation tool, i-DME also proposes repairs on the diagnostic/system model that achieve better performance. Importantly, i-DME also pre-accredits the diagnostic model embedded in software systems and derives the corresponding necessary accreditation requirements for the embedded system. This facilitates isolating the root cause if the model or the code within which the model is embedded is the cause of degraded performance in the case of embedded systems. This is necessary as NASA standards, viz. NASA-STD-7009 and NPR 7150.2A, have restrictions to accredit all the embedded systems. For this purpose, process to translate knowledge from model accreditation to

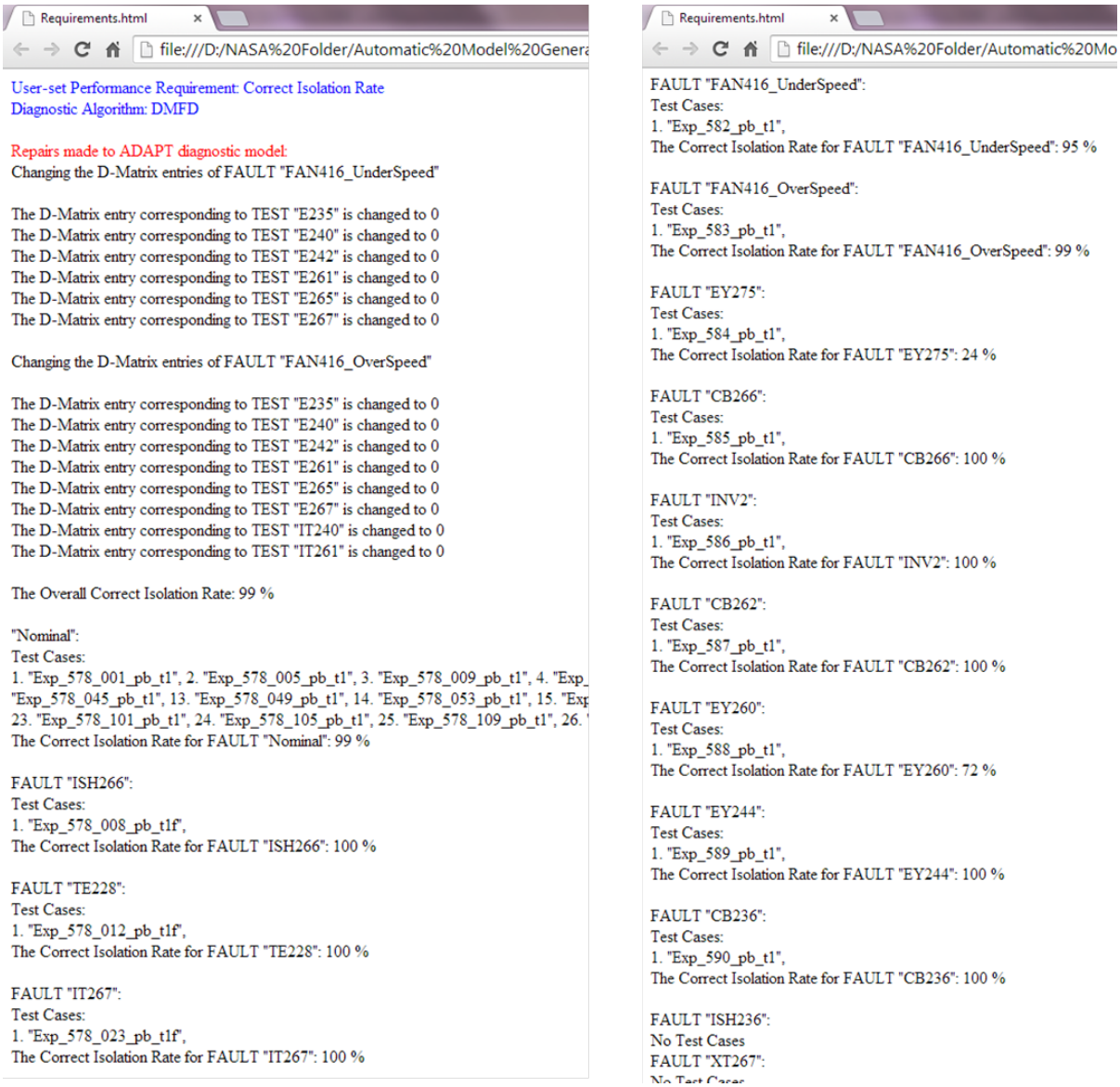

Figure 3. Reporting of accreditation requirements for embedded ADAPT system

embedded system accreditation as requirements is defined. i-DME automatically generates for verification and validation requirements, thus making it possible to accredit even very large-scale embedded diagnostic systems. In the future, we will explore for uncertainty requirements (requirements 4.4.7 – 4.4.9 in NASA-STD-7009) and credibility assessment score that are necessary for accrediting embedded systems and implement them in i-DME.

REFERENCES

CAIB (2003). Columbia Accident Investigation Board Report. . vol. 1.

Daigle, M., Roychoudhury, I., Biswas, G., & Koutsoukos, X (2010). An event-based approach to distributed diagnosis of continuous systems. *Proceedings of the 21st International Workshop on Principles of Diagnosis*, pp. 15-22.

Kodali, A., Robinson, P., & Patterson-Hine, A. (2013). A framework to debug diagnostic matrices. *Annual Conference of the Prognostics and Health Management Society 2013*, October 14 - 17, New Orleans, LO.

Luo, J., Tu, H., Pattipati, K., Qiao, L., & Chigusa, S. (2006). Graphical models for diagnostic knowledge representation and inference. *IEEE Instrum. Meas. Mag.*, vol. 9, no. 4, pp. 45–52.

NASA-STD-7009 (2008). Standards for models and simulations. NASA, https://standards.nasa.gov/documents/viewdoc/3315599/3315599.

NASA software engineering handbook (2013). *NASA Technical Handbook*. NASA, http://swehb.nasa.gov/display/7150/7.15+-+Relationship+Between+NPR+7150.2+and+NASA-STD-7009#_ tabs-1.

NPR 7150.2A (2009). NASA software engineering requirements. NASA, http://nodis3.gsfc.nasa.gov/displayDir.cfm?t=NPR&c=7150&s=2.

Poll, S., Patterson-Hine, A., Camisa, J., Garcia, D., Hall, D., Lee, C., Mengshoel, O., Neukom, C., Nishikawa, D., Ossenfort, J., Sweet, A., Yentus, S., Roychoudhury, I., Daigle, M., Biswas, G., & Koutsoukos, X. (2007). Advanced diagnostics and prognostics testbed. *In Proc. DX'07*, pp. 178–185.

Qualtech Systems Inc., www.teamqsi.com.

Sabetzadeh, M., Nejati, S., A., Briand, L., & Mills, A. E. (2011). Using SysML for modeling of safety-critical software–hardware interfaces: Guidelines and industry Experience. *IEEE 13th International Symposium on High-Assurance Systems Engineering*.

Sheppard, J. W., & Simpson, W. R. (1991). A mathematical model for integrated diagnostics. *IEEE Design and Test of Computers*, vol. 8, no. 4, pp. 25 – 38.

Sheppard, J. W., & Simpson, W. R. (1992). Applying testability analysis for integrated diagnostics. *IEEE Design and Test of Computers*, vol. 9, no. 3, pp. 65 – 78.

Sheppard, J. W., & Simpson, W. R. (1993). Performing effective fault isolation in integrated diagnostics. *IEEE Design and Test of Computers*, vol. 10, no. 2, pp. 78 – 90.

Sheppard, J. W., & Simpson, W. R. (1998). Managing conflicts in system diagnostics. *IEEE Computer*, vol. 31, no. 3, pp. 69 – 76.

Simpson, W. R., & Sheppard, J. W. (1991). System complexity and integrated diagnostics. *IEEE Design and Test of Computers*, vol. 8, no. 3, pp. 16 -30.

Simpson, W. R., & Sheppard, J. W. (1992). System testability assessment for integrated diagnostics. *IEEE Design and Test of Computers*, vol. 9, no. 1, pp. 40 -54.

Simpson, W. R., & Sheppard, J. W. (1993). Fault isolation in an integrated diagnostics. *IEEE Design and Test of Computers*, vol. 10, no. 1, pp. 52 -66.

Singh, S., Kodali, A., Choi, K., Pattipati, K., Namburu, S., Chigusa, S., Prokhorov, D.V., & Qiao, L. (2009). Dynamic multiple fault diagnosis: Mathematical formulations and solution techniques. *IEEE Trans. Syst., Man, Cybern*. A, vol. 39, no. 1, pp. 160–176.

Swenson, Jr., L. S., & Grimwood, J. M. (1989). *This new ocean: A history of project Mercury*. Published as NASA Special Publication-4201 in the NASA History Series.

Vaandrager, F. W. (2006). Does it pay-off? model-based verification and validation of embedded systems!. *In F. A. Karelese (editor), PROGRESS White Papers*.